Symmetry and Bandstructure

Hosson, J.Th.M. De

Published in:
Computer Physics Communications

DOI:
10.1016/0010-4655(75)90091-0

Publication date:
1975

Computer Physics Communications 10 (1975) 67–69

SYMMETRY AND BANDSTRUCTURE

J.Th.M. DE HOSSON
Laboratorium voor Fysische Metaalkunde, Materials Science Centre, Paddepoel, Groningen, The Netherlands

Received 17 December 1974

ADAPTATION SUMMARY

Title of adaptation: SYMMETRY AND BANDSTRUCTURE

Adaptation number: 0001

Program obtainable from: CPC Program Library, Queen's University of Belfast, N. Ireland (see application form in this issue).

Reference to original program: *Catalogue number*: ACMJ; *Title*: SYMMETRIZED APW; *Ref. in CPC*: 4 (1972) 361

Authors of original program: V. Hoffstein, D.K. Ray and M. Belakhovsky

Computer: Control Data CYBER 74

Operating system: SCOPE 3.4.1. level 373

Programming language used in adapted program: FORTRAN

High speed storage required: 70742 words

No. of bits in a word: 60

No. of cards required to effect adaptation: 39

Card punching code: IBM (029)

Nature of physical problem
The program ACMJ (1) calculates the energy bands and electronic structure of cubic lattices with no more than two different atoms per unit cell.

Method of solution
The program was designed to run on an IBM computer and this adaptation converts it for use on CDC computers. The changes are also suitable for the program ACME [2].

The method of calculation of the bandstructure is the Symmetry Augmented Plane-Wave method (SAPW) which makes use of the point group symmetry of the hamiltonian.

Restrictions on the complexity of the problem
Only for cubic lattices.

Typical running time
For a 3 X 3 matrix eigenvalue problem calculated with ACMJ: 17 seconds with the FTN-compiler and 12 seconds with the RUN-compiler.

Unusual features of the program
None.

References
[1] V. Hoffstein, D.K. Ray, M. Belakhosvky, Computer Phys. Commun. 4 (1972) 361.
[2] V. Hoffstein and O. Moller, Computer Phys. Commun. 2 (1971) 17.

LONG WRITE-UP

1. Introduction

The programs for calculating energy bands and electronic structure of solids, written by Hoffstein et al. [1–3] are designed to run on IBM computer systems. This adaptation modifies the program SYMMETRIZED APW [3], catalogue number ACMJ, to run on a CDC computer.

The main changes are to the FUNCTION CONV (IS) which computes a real mode equivalent for the alphameric word IS. As it stands this function could not be used on a CDC computer as it manipulates and compares characters in a word using division and subtraction. Apart from the effect of the number of bits in a character and in a word being different on the two computers, on a CDC the arithmetic processes would result in a loss of characters. This adaptation makes use of the CDC intrinsic function SHIFT and masking expressions.

The function SHIFT shifts a variable of any type over an integer number of bit positions, left or right circular. It is an intrinsic function with the FTN-compiler. At CDC installations using the RUN-compiler the shifting function is called ISHIFT. However, in case it is unavailable a version written in the COMPASS [4] language is included in the adaptation deck.

The program SYMMETRY AND BAND STRUCTURE 2 [2], catalogue number ACME, also contains FUNCTION CONV which can be altered by the appropriate cards from this adapation deck without changing the main program or the data.

2. The subprogram CONV(IS)

The subprogram CONV (IS) performs the conversion of a four character alphameric word IS into its floating point equivalent. It is designed to convert the elements and column headings of the Luehrmann tables [5] which are read in by the program in Format A4. The Luehrmann tables, calculated by the projection operator method, contain the coefficients for forming linear combinations of plane waves both with and without spin, symmetrized with respect to all the necessary subgroups of all the symmorphic space groups. Three digit integers represent the permutation associated with a column heading. The entries of the tables are one or two digit integers or the characters A, B, C, which are used to represent $\frac{1}{2}\sqrt{3}$, 0.5 and 1.5 for symmorphic space groups. The possibilities for the form of IS are listed in table 1.

Table 1
Correlation for IS and its real mode number

IS	Real mode number computed from IS
$bI_3I_2I_1$	$100I_3 + 10I_2 + I_1$
$b \pm I_2I_1$	$\pm (10I_2 + I_1)$
$bb \pm I_1$	$\pm I_1$
bb b b	0

The conversion of the digits from character form into a real number is performed by comparison with a look-up table SYM consisting of all possible characters, the numerical value of a character being related to its position in the table. The elements of SYM(J) are in the form *bbbN* where *b* stands for an alphameric blank symbol and *N* for a character.

Suppose IS contains the column heading, *b*615. The character information is stored in six bit display code with ten characters per sixty bit word. The quantity b615 is stored left in the word, the remainder being filled with blanks. The internal representation (octal) of IS is given by (see table 2):

IS = 55 41 34 40 55 55 55 55 55 55 .

In order to make the comparison with the elements of SYM each integer character in turn must first be moved to the fourth character position (positions 19–24 of the bit pattern). On the CDC computer this can be achieved by the intrinsic function SHIFT(IS, *n*) which shifts IS over *n* bit positions, left circular if *n* is positive

Table 2
Table of the representations character – bit – display code

Character	Bit pattern	Display code
1	011 100	34
5	100 000	40
6	100 001	41
+	100 101	45
–	100 110	46
blank	101 101	55

and right circular if n is negative.

With the following statements

I2 = SHIFT(IS, −6), I3 = SHIFT(IS, −12),

one obtains in I2 and I3 in octal representation:

I2 = 77 55 41 34 40 55 55 55 55 55,

I3 = 77 77 55 41 34 40 55 55 55 55.

The ones in the left-most positions are caused by sign extension. It is then possible to isolate each character by masking with an integer, M, which has ones in the bit positions 19–24 and zeros elsewhere. The masking expression .AND. forms the bit-by-bit logical product. For example I2.AND.M gives (only the first thirty bits are written out):

I_2:	111 111 101 101 100 001 011 100 100 000 ... ...
M:	000 000 000 000 000 000 111 111 000 000 ... ...
Result:	000 000 000 000 000 000 011 100 000 000 ... ...

The same masking operation is performed on the elements of the array SYM so that the characters in the look-up table are also packed with zeros. A direct logical comparison can then be made between the isolated characters and the table as in the original program.

Acknowledgements

I wish to thank Mr. A.C.M. van Swieten for helpful comments. The research program for electronic structure calculations is sponsored by the Foundation for Fundamental Research on Matter (F.O.M.) and The T.N.O. Institute for Metals.

References

[1] V. Hoffstein and O. Moller, Computer Phys. Commun. 2 (1971) 17.
[2] V. Hoffstein and O. Moller, Computer Phys. Commun. 2 (1971) 26.
[3] V. Hoffstein, D.K. Ray and M. Belakhovsky, Computer Phys. Commun. 4 (1972) 361.
[4] Control Data, Compass Reference Manual, CYBER 70 series version 3, p. 8–28, p. 8–32.
[5] A.W. Luehrmann, Advan. Phys. 17 (1968) 65, 1.